**Jack**

* ~~Set up MongoDB on database server & give login to Theja~~
* ~~Base calculation options for webpage off of empty method stubs~~
* Implement AngularJS to make our page truly single page

**Theja**

* ~~Research MongoDB~~
* ~~Create some of our tables in MongoDB as collections~~
* Help Ibrahim and Kevin with the relationships in our MySQL database

**Swaraj**

* ~~Help Theja automatically populate MongoDB data structures~~
* Help with populating MySQL tables (has experience with maintaining relationships)

**Satabdi**

* ~~Audit the team based on our requirements document~~
* ~~Make sure we implement functions/rules/use cases that we specified in the documents~~
* ~~Bring up any functions that we might want to delete from the documentation~~
* ~~Research Traceability matrix~~

**Ibrahim**

* ~~Create Java code to automatically generate tables~~
* ~~Use the code to populate tables~~

**Kevin**

* ~~Help Ibrahim fill our tables with data~~
* ~~We need a few companies with lots of employees to start~~

**Marc**

* ~~Research JUnit testing and Jmeter performance testing~~
* Write some JUnit tests that we can run on our Java code
* Make a new branch for JUnit testing and push it to bitbucket

Friday meeting at 4pm

* Fill out method stubs for querying different employees based on rank
* Split up the different services across different servers
* Documentation needs to be proofread
* JUnit tests need to be integrated into project
* JMeter needs to be integrated to stresstest (creates a bunch of clients)
* Java code to have command line interface
* New stub for generating the new company with an unused company ID
* Write MongoDBService to query mongo tables
* Theja integrate MongoDBGenerator into main branch (called MongoUtilities)
* Work as a group to finish the documentation on Sunday
* Link MongoDB database with SQL data, check transportation costs and check mongodb to see if employees could from home
* Make sure database is 300mb
* Get 1000 pictures for all employees
* (maybe) Scaffolding for Website - builds JSP pages base off of Domain Objects
* Requirements update ~ 2 hours
* Design document update ~ 2 hours
* Database Design Update ~ 1 hour
* Traceabilitiy Matrix ~ 3 hours
* Installation documentation ~ 2 hours
* Configuration documentation ~ 1 hour
* JUnit Tests ~5 hours
* JMeter Performance Testing ~ 4 hours
* Source code package ~ 1 hour
* Front end GUI update ~ 3 hours
* Populate the whole damn database ~ 3 hours
* Actually use NoSQL for some table ~ 4 hours
* More methods to use rest of data ~ 3 hours
* GUI development ~ 2 hours
* Extra credit, have one large company that automatically starts up extra VM once they log into the service ~ 5 hours
* Use open source SDN and write a control program that plays the role of a load balancer by directing requests to different destinations based on their busy status.